Package ‘bioCancer’

February 28, 2024

Title Interactive Multi-Omics Cancers Data Visualization and Analysis

Version 1.30.8

Date 2024-02-14

Description This package is a Shiny App to visualize and analyse interactively Multi-Assays of Cancer Genomic Data.

Depends R (>= 3.6.0), radiant.data (>= 0.9.1), cBioPortalData, XML(>= 3.98)

Imports R.oo, R.methodsS3, DT (>= 0.3), dplyr (>= 0.7.2), tidyr, shiny (>= 1.0.5), AlgDesign (>= 1.1.7.3), import (>= 1.1.0), methods, AnnotationDbi, shinythemes, Biobase, geNetClassifier, org.Hs.eg.db, org.Bt.eg.db, DOSE, clusterProfiler, reactome.db, ReactomePA, DiagrammeR(<= 1.01), visNetwork, htmlwidgets, plyr, tibble, GO.db

Suggests BiocStyle, prettydoc, rmarkdown, knitr, testthat (>= 0.10.0)

VignetteBuilder knitr

URL https://kmezhoud.github.io/bioCancer/

BugReports https://github.com/kmezhoud/bioCancer/issues

License AGPL-3 | file LICENSE

LazyData true

biocViews GUI, DataRepresentation, Network, MultipleComparison, Pathways, Reactome, Visualization, GeneExpression, GeneTarget

RoxygenNote 7.3.1

Encoding UTF-8

git_url https://git.bioconductor.org/packages/bioCancer

git_branch RELEASE_3_18

git_last_commit 94d43a0

Repository Bioconductor 3.18

Date/Publication 2024-02-28

Author Karim Mezhoud [aut, cre]

Maintainer Karim Mezhoud <kmezhoud@gmail.com>
R topics documented:

.dbEscapeString .................................................. 3
.getTableName ...................................................... 4
.pickRef ............................................................... 4
.AnnotationFuncs ................................................... 5
.attrIColorGene ..................................................... 6
.attrIColorValue ................................................... 7
.attrIColorVector ................................................... 8
.attriShape2Gene .................................................. 9
.attriShape2Node .................................................. 9
.bioCancer ........................................................... 10
.CGDS ................................................................. 10
.checkDimensions .................................................. 11
.coffeewheel ......................................................... 12
.coffeewheelOutput ................................................ 12
.displayTable ....................................................... 13
.Edges_Diseases_obj ................................................ 14
.epiGenomics ....................................................... 15
.findPhantom ....................................................... 15
.getEvidenceCodes ................................................ 16
.getFreqMutData ................................................... 16
.getGenesClassification .......................................... 17
.getListProfData ................................................... 18
.getList_Cases ..................................................... 19
.getList_GenProfs ................................................. 20
.getOrthologs ....................................................... 20
.getProfData ........................................................ 22
.getSequensed_SampleSize ....................................... 23
.mapLists ............................................................. 24
.metabologram ...................................................... 25
.metabologramOutput ............................................. 26
.Mutation_obj ....................................................... 26
.Node_df_FreqIn .................................................... 27
.Node_Diseases_obj ................................................ 28
.Node_obj_CNA_ProfData .......................................... 29
.Node_obj_FreqIn ................................................... 29
.Node_obj_Met_ProfData .......................................... 30
.Node_obj_mRNA_Classifier ...................................... 31
.pickGO .............................................................. 32
.pickRefSeq ........................................................ 33
.removeNAs .......................................................... 34
.renderCoffeewheel ................................................ 35
.renderMetabologram ............................................. 36
.reStrColorGene .................................................... 36
.reStrDimension .................................................... 37
.reStrDisease ....................................................... 38
.returnTextAreaInput .............................................. 39
.dbEscapeString

Description

Does not escape strings, but raises an error if any character except normal letters and underscores are found in the string.

Usage

.dbEscapeString(str, raise.error = TRUE)

Arguments

str String to test
raise.error Logical, whether to raise an error or not.

Value

Invisible logical
.getTableName

*Gets the table name from the INPARANOID style genus names.*

**Description**

Gets the table name from the INPARANOID style genus names.

**Usage**

```
.getTableName(genus)
```

**Arguments**

- **genus**: 5 character INPARANOID genus name, such as "BOSTA", "HOMSA" or "MUSMU".

**Value**

Table name for genus.

**Author(s)**

Stefan McKinnon Edwards <stefanm.edwards@agrsci.dk>

**References**

[https://www.bioconductor.org/packages/release/bioc/html/AnnotationDbi.html](https://www.bioconductor.org/packages/release/bioc/html/AnnotationDbi.html)

---

.pickRef

*Secret function that does the magic for pickRefSeq.*

**Description**

Do not use it, use `pickRefSeq`!

**Usage**

```
.pickRef(l, priorities, reduce = c("all", "first", "last"))
```

**Arguments**

- **l**: List.
- **priorities**: How to prioritize.
- **reduce**: How to reduce.
Value
List.

Note
Hey, you found a secret function! Keep it that way!

Author(s)
Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

See Also
pickRefSeq

Description

<table>
<thead>
<tr>
<th></th>
<th>AnnotationFuncs</th>
<th>Annotation translation functions</th>
</tr>
</thead>
</table>

| Package:                | AnnotationFuncs |                                   |
| Type:                   | Package         |                                   |
| Version:                | 1.3.0           |                                   |
| Date:                   | 2011-06-10      |                                   |
| License:                | GPL-2           |                                   |
| LazyLoad:               | yes             |                                   |

Details
Functions for handling translations between different identifiers using the Biocore Data Team data-packages (e.g. org.Bt.eg.db). Primary functions are translate for translating and getOrthologs for efficient lookup of homologs using the Inparanoid databases. Other functions include functions for selecting Refseqs or Gene Ontologies (GO).

Author(s)
Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

References
### attriColorGene

Attribute Color to Gene

#### Description

Attribute Color to Gene

#### Usage

attriColorGene(df)

#### Arguments

- **df**: data frame with mRNA or CNA or mutation frequency or methylation (numeric). Without sampleID column.

#### Value

A list colors for every gene

#### Examples

```r
cgds <- cBioPortal(  hostname = "www.cbioportal.org",  protocol = "https",  api = "/api/v2/api-docs")  ## Not run:  getDataByGenes( api = cgds,  studyId = "gbm_tcga_pub",  genes = c("NF1", "TP53", "ABL1"),  by = "hugoGeneSymbol",  molecularProfileIds = "gbm_tcga_pub_mrna")```
### attriColorValue

#### Description

Attribute Color to Value

#### Usage

```r
attriColorValue(Value, df, colors=c(a,b,c),feet)
```

#### Arguments

- **Value**
  - integer
- **df**
  - data frame with numeric values
- **colors**
  - a vector of 5 colors
- **feet**
  - the interval between two successive colors in the palette (0.1)

#### Value

Hex Color Code

#### Examples

```r
cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)
## Not run:
getDataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
  molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)
```
### attriColorVector

**Attribute color to a vector of numeric values**

#### Description

Attribute color to a vector of numeric values

#### Usage

```R
attriColorVector(Value, vector, colors=c(a,b,c), feet)
```

#### Arguments

- **Value**: numeric
- **vector**: A vector of numeric data
- **colors**: 3 colors
- **feet**: An interval between two numeric value needed to change the color

#### Value

A vector of colors

#### Examples

```R
cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)
## Not run:
getDataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
  molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)
```
**attriShape2Gene**  
*Attribute shape to nodes*

**Description**

Attribute shape to nodes

**Usage**

`attriShape2Gene(gene, genelist)`

**Arguments**

- `gene`: Gene symbol
- `genelist`: Gene list

**Value**

A character "BRCA1[shape = 'circle', "

**Examples**

```r
how <- "runManually"
## Not run:
Genelist <- whichGeneList("73")
attriShape2Gene("P53", Genelist)
attriShape2Gene("GML",Genelist)
## End(Not run)
```

**attriShape2Node**  
*Attributes shape to Nodes*

**Description**

Attributes shape to Nodes

**Usage**

`attriShape2Node(gene, genelist)`

**Arguments**

- `gene`: symbol "TP53"
- `genelist`: a vector of gene symbol
Value

A data frame with egdes attributes

Examples

```r
GeneList <- c("DKK3", "NBN", "MYO6", "TP53", "PML", "IFI16", "BRCA1")
NodeShape <- attriShape2Gene("DKK3", GeneList)
```

---

**bioCancer**

*Launch bioCancer with default browser*

---

**Description**

The Main function to run bioCancer App

**Usage**

```r
bioCancer()
```

**Value**

web page of bioCancer Shiny App

**Examples**

```r
ShinyApp <- 1
## Not run:
bioCancer()
## End(Not run)
```

---

**CGDS**

*CGDS connect object to cBioPortal*

---

**Description**

Creates a CGDS connection object from a CGDS endpoint URL. This object must be passed on to the methods which query the server.

**Usage**

```r
CGDS(url, verbose=FALSE, ploterrormsg='', token=NULL)
```
checkDimensions

Arguments

url          A CGDS URL (required).
verbose      A boolean variable specifying verbose output (default FALSE)
ploterrormsg An optional message to display in plots if an error occurs (default '')
token        An optional 'Authorization: Bearer' token to connect to cBioPortal instances
             that require authentication (default NULL)

checkDimensions  Check which Cases and genetic profiles are available for selected study

Description

Check which Cases and genetic profiles are available for selected study

Usage

checkDimensions(StudyID)

Arguments

StudyID      Study reference using cBioPortal index

Value

A data frame with two column (Cases, Genetic profiles). Every row has a dimension (CNA, mRNA...). The data frame is filled with yes/no response.

Examples

cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)
## Not run:
getDataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
  molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)
coffeewheel

This is an htmlwidgets-based visualization tool for hierarchical data. It is zoomable, meaning that you can interact with the hierarchy and zoom in/out accordingly.

Description

This is an htmlwidgets-based visualization tool for hierarchical data. It is zoomable, meaning that you can interact with the hierarchy and zoom in/out accordingly.

Usage

coffeewheel(treeData, width=600, height=600, main=", partitionAttribute="value")

Arguments

treeData A hierarchical tree data as in example
width 600
height 600
main Title
partitionAttribute "value"

Value

A circular layout with genetic profile.

Examples

How <= "runManually"
## Not run:
coffeewheel(treeData = sampleWheelData)

## End(Not run)

coffeewheelOutput

Widget output function for use in Shiny

Description

Widget output function for use in Shiny

Usage

coffeewheelOutput(outputId, width=700, height=700)
### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>outputId</td>
<td>id</td>
</tr>
<tr>
<td>width</td>
<td>700</td>
</tr>
<tr>
<td>height</td>
<td>700</td>
</tr>
</tbody>
</table>

### Value


### Examples

```r
# Not run:
coffeewheel(treeData = sampleWheelData)
```

---

**displayTable**  
Display dataframe in table using DT package

### Description

Display dataframe in table using DT package

### Usage

displayTable(df)

### Arguments

- **df**: a dataframe

### Value

A table

### Examples

```r
cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)
# Not run:
getDataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
```
molecularProfileIds = "gbm_tcga_pub_mrna"

## End(Not run)

---

**Edges_Diseases_obj**

get Edges dataframe for Gene/Disease association from geNetClassifier

**Description**

get Edges dataframe for Gene/Disease association from geNetClassifier

**Usage**

```r
Edges_Diseases_obj(genesclassdetails)
```

**Arguments**

- `genesclassdetails`:
  a dataframe from geNetClassifier

**Value**

A data frame with egdes attributes

**Examples**

```r
GenesClassDetails <- structure(list(Genes = c("FANCF", "MLH1", "MSH2", "ATR", "PARP1", "CHEK2", "RAD51"), ranking = c(1L, 1L, 1L, 2L, 3L, 1L, 2L), class = c("brca_tcga", "gbm_tcga", "lihc_tcga", "lihc_tcga", "lihc_tcga", "lusc_tcga", "lusc_tcga"), postProb = c(1, 0.99, 1, 0.99, 0.99, 1, 0.98), exprsMeanDiff = c(180, 256, -373, -268, -1482, 258, 143), exprsUpDw = c("UP", "UP", "DOWN", "DOWN", "UP", "UP")), .Names = c("Genes", "ranking", "class", "postProb", "exprsMeanDiff", "exprsUpDw"), class = "data.frame", row.names = c(NA, -7L))

Ed_Diseases_obj <- Edges_Diseases_obj(genesclassdetails=GenesClassDetails)
```
**epiGenomics**

Default dataset of bioCancer

**Description**

Default dataset of bioCancer

**Usage**

epiGenomics

**Format**

An object of class data.frame with 48 rows and 7 columns.

**Author(s)**

Karim Mezhoud <kmezhoud@gmail.com>

---

**findPhantom**

Check if PhantomJS is installed. Similar to webshot

**Description**

Check if PhantomJS is installed. Similar to webshot

**Usage**

findPhantom()

**Value**

Logic object

**Examples**

```r
How <- "runManually"
## Not run:
findPhantom()
## End(Not run)
```
**getEvidenceCodes**  
*Returns GO evidence codes.*

**Description**

Returns GO evidence codes.

**Usage**

getEvidenceCodes()

**Value**

Matrix of two columns, first column with codes, second column with description of codes.

**Author(s)**

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

**References**

?org.Bt.egGO

**See Also**

pickGO

**Examples**

getEvidenceCodes()

---

**getFreqMutData**  
*get mutation frequency*

**Description**

get mutation frequency

**Usage**

getFreqMutData(list, geneListLabel)

**Arguments**

- list  
a list of data frame with mutation data. Each data frame is for one study
- geneListLabel  
file name of geneList examples: "73"
getGenesClassification

Value

a data frame with mutation frequency. gene is in rows and study is in column

Examples

cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)

## Not run:
getDataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
  molecularProfileIds = "gbm_tcga_pub_mrna"
)

## End(Not run)

getGenesClassification

get genes classification

Description

get genes classification

Usage

getGenesClassification(checked_Studies, GeneList,
  samplesize, threshold, listGenProfs, listCases)

Arguments

checked_Studies   checked studies
GeneList          gene list
samplesize        sample size
threshold         p-value threshold
listGenProfs      list of genetic profiles
listCases         list of cases

Value

A table with genes classed by study
 getListProfData

Examples

cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)

## Not run:
getDataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
  molecularProfileIds = "gbm_tcga_pub_mrna"
)

## End(Not run)

ggetListProfData  
get list of data frame with profiles data (CNA,mRNA, Methylation, Mutation...)

Description

get list of data frame with profiles data (CNA,mRNA, Methylation, Mutation...)

Usage

ggetListProfData(checked_Studies, geneListLabel)

Arguments

checked_Studies  
checked studies in corresponding panel (input$StudiesIDCircos, input$StudiesIDReactome).

geneListLabel  
The label of GeneList. There are three cases: "Genes" user gene list, "Reactome_GeneList" GeneList plus genes from reactomeFI "file name" from Examples

Value

A LIST of profiles data (CNA, mRNA, Methylation, Mutation, miRNA, RPPA). Each dimension content a list of studies.

Examples

cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)
getList_Cases

get list of cases of each selected study in Classifier panel

Description

get list of cases of each selected study in Classifier panel

Usage

ggetList_Cases(checked_Studies)

Arguments

checked_Studies

cHECKED studies

Value

A list of cases

Examples

cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)
## Not run:
getDataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
  molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)

## Not run:
getDataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
  molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)
### getList_GenProfs

**get list of genetic profiles of each selected study in Classifier panel**

**Description**

get list of genetic profiles of each selected study in Classifier panel

**Usage**

`getList_GenProfs(checked_Studies)`

**Arguments**

`checked_Studies`

checked studies

**Value**

A list of genetics profiles

**Examples**

```r
cgds <- cBioPortal(
    hostname = "www.cbioportal.org",
    protocol = "https",
    api = "/api/v2/api-docs"
)
## Not run:
getDataByGenes( api = cgds,
    studyId = "gbm_tcga_pub",
    genes = c("NF1", "TP53", "ABL1"),
    by = "hugoGeneSymbol",
    molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)
```

### getOrthologs

**Performs quicker lookup for orthologs in homologe data packages**

**Description**

Using the INPARANOID data packages such as hom.Hs.inp.db is very, very slow and can take up to 11 min (on this particular developers workstation). This function introduces a new method that can do it in just 20 seconds (on the developers workstation). In addition, it includes options for translating between different identifiers both before and after the mapping.
getOrthologs

Usage

getOrthologs(
  values,
  mapping,
  genus,
  threshold = 1,
  pre.from = NULL,
  pre.to = NULL,
  post.from = NULL,
  post.to = NULL,
  ...
)

Arguments

values Vector, coerced to character vector, of values needed mapping by homology.
mapping Homology mapping object, such as `hom.Hs.inpBOSTA` or `revmap(hom.Hs.inpBOSTA)`.
genus Character vector. 5 character INPARANOID style genus name of the mapping object, e.g. ‘BOSTA’ for both `hom.Hs.inpBOSTA` and `revmap(hom.Hs.inpBOSTA)`.
threshold Numeric value between 0 and 1. Only clustered homologues with a pairwise score above the threshold is included. The native implementation has this set to 1.
pre.from Mapping object if values needs translation before mapping. E.g. values are entrez and `hom.Hs.inpBOSTA` requires ENSEMBLPROT, `hom.Hs.inpAPIME` requires Refseq (?). Arguments from and to are just like in `translate`.
pre.to Second part of translation before mapping.
post.from Translate the result from homology mapping to a desired id; just like in `translate`.
post.to Second part of translation after mapping.
... Additional arguments sent to `translate`.

Value

List. Names of list corresponds to values, except those that could not be mapped nor translated. Entries are character vectors.

Author(s)

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

References

?hom.Hs.inp.db - https://inparanoidb.sbc.su.se/


### See Also

`translate`, `.getTableName`, `mapLists`

### Examples

```r
tmp <- 1
```

---

```r
gprofData  
```

### Description

search and get genetic profiles (CNA, mRNA, Methylation, Mutation...)

### Usage

```r
gprofData(study, genProf, listGenProf, GeneList, Mut)
```

### Arguments

- **study**: Study ID
- **genProf**: Genetic Profile id (cancer_study_id_[mutations, cna, methylation, mrna ]).
- **listGenProf**: A list of Genetic Profiles for one study.
- **GeneList**: A list of genes
- **Mut**: Condition to set if the genetic profile is mutation or not (0,1)

### Details

See [https://github.com/kmezhoud/bioCancer/wiki](https://github.com/kmezhoud/bioCancer/wiki)

### Value

A data frame with Genetic profile
getSequensed_SampleSize

get samples size of sequenced genes

Description

get samples size of sequenced genes

Usage

getSequensed_SampleSize(StudiesID)

Arguments

StudiesID Studies ID as a vector

Value

dataframe with sample size for each selected study.

Examples

## Not run:
sampleSize <- getSequensed_SampleSize(input$StudiesIDCircos)

## End(Not run)
mapLists

Replaces contents of list A with elements of list B

Description

Combines two lists, A and B, such that names(A) are preserved, mapping to the values of B, using names(B) as look up. I.e. replaces values in A with values in B, using names(B) as look up for values in A. Once more? See examples. NB! None-mapped entries are returned as NA, but can be removed using removeNAs.

Usage

mapLists(A, B, removeNAs = TRUE)

Arguments

A List, elements are coerced to character for mapping to B.
B List.
removeNAs Boolean, whether to remove the NAs that occur because an element was not found in B.

Value

List.

Author(s)

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

See Also

removeNAs

Examples

A <- list('a1'='alpha', 'a2'='beta', 'a3'=c('gamma','delta'))
B <- list('alpha'='b1', 'gamma'=c('b2', 'b3'), 'delta'='b4')
mapLists(A, B)
metabologram

Circular plot of hierarchical data of genetic profile.

Description

Circular plot of hierarchical data of genetic profile.

Usage

```r
metabologram(treeData, width=600, height=600, main="" , showLegend=FALSE,
             legendBreaks=NULL, legendColors=NULL, fontSize=12,
             legendText="Legend")
```

Arguments

- `treeData`: A hierarchical tree data as in example
- `width`: 600
- `height`: 600
- `main`: Title
- `showLegend`: FALSE
- `legendBreaks`: NULL
- `legendColors`: NULL
- `fontSize`: 12
- `legendText`: Legend

Value

A circular layout with genetic profile.

See Also

https://github.com/armish/metabologram

Examples

```r
How <- "runManually"
## Not run:
metabologram(treeData = sampleWheelData, width=600,
             height=600, main="title", showLegend = TRUE, fontSize = 10,
             legendBreaks=c("NA","Min","Negative","0","Positive","Max"),
             legendColors=c("black","blue","cyan","white","yellow","red"),
             legendText="Legend")
## End(Not run)
```
metabologramOutput  Widget output function for use in Shiny

Description
Widget output function for use in Shiny

Usage
metabologramOutput(outputId, width = 600, height = 500)

Arguments
outputId id
width 600
height 600

Value
A circular plot with genetic profile in Shiny App.

Examples
## Not run:
library(bioCancer)
bioCancer::metabologram(treeData = sampleMetabologramData)
## End(Not run)

Mutation_obj  Attribute mutation frequency to nodes

Description
Attribute mutation frequency to nodes

Usage
Mutation_obj(list,FreqMutThreshold, geneListLabel)

Arguments
list A list of data frame with mutation data. Each data frame to study
FreqMutThreshold threshold Rate of cases (patients) having mutation (0-1).
geneListLabel file name of geneList examples: "73"
Node_df_FreqIn

Value
A data frame with mutation frequency. Each column corresponds to a study.

Examples

cgds <- cBioPortal(
    hostname = "www.cbioportal.org",
    protocol = "https",
    api = "/api/v2/api-docs"
)
## Not run:
getDataByGenes( api = cgds, 
    studyId = "gbm_tcga_pub", 
    genes = c("NF1", "TP53", "ABL1"), 
    by = "hugoGeneSymbol", 
    molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)

Node_df_FreqIn

Attributes size to Nodes depending on number of interaction

Description
Attributes size to Nodes depending on number of interaction

Usage
Node_df_FreqIn(genelist, freqIn)

Arguments

| genelist | a vector of genes |
| freqIn   | dataframe with Node interaction frequencies |

Value
A data frame with nodes size attributes

Examples
Node_df_FreqIn
## Not run:
r_data <- new.env()
r_data[['FreqIn']] <- structure(list(Genes = c("ATM", "ATR", "BRCA1", "BRCA2", "CHEK1", "CHEK2", "FANCF", "MDC1", "RAD51"), FreqSum = c(0.04, 0.05, 0.05, 0.03, 0.05, 0.03, 0.03, 0.03, 0.02)), .Names = c("Genes", "FreqSum"),...
Node_Diseases_obj

Attributes color and shape to Nodes of Diseases

Description

Attributes color and shape to Nodes of Diseases

Usage

Node_Diseases_obj(genesclassdetails)

Arguments

genesclassdetails

  a dataframe from geNetClassifier function

Value

A data frame with nodes Shapes and colors

Examples

GenesClassDetails <- structure(list(Genes = c("FANCF", "MLH1", "MSH2", "ATR", "PARP1", "CHEK2", "RAD51"), ranking = c(1L, 1L, 1L, 2L, 3L, 1L, 2L), class = c("brca_tcga", "gbm_tcga", "lihc_tcga", "lihc_tcga", "lusc_tcga", "lusc_tcga"), postProb = c(1, 0.99, 1, 0.99, 0.99, 1, 0.98), exprsMeanDiff = c(180, 256, -373, -268, -1482, 258, 143), exprsUpDw = c("UP", "UP", "DOWN", "DOWN", "UP", "UP")), .Names = c("Genes", "ranking", "class", "postProb", "exprsMeanDiff", "exprsUpDw"), class = "data.frame", row.names = c(NA,-7L))

Node_Diseases_df <- Node_Diseases_obj(genesclassdetails= GenesClassDetails)
**Node_obj_CNA_ProfData**  
*Attribute CNA data to node border*

**Description**  
Attribute CNA data to node border

**Usage**  
Node_obj_CNA_ProfData(list)

**Arguments**  
- **list**: A list of data frame with CNA data. Each data frame corresponds to a study.

**Value**  
A data frame with node border attributes

**Examples**
```r
cgds <- cBioPortal(  
  hostname = "www.cbioportal.org",  
  protocol = "https",  
  api = "/api/v2/api-docs"  
)
## Not run:  
getDataByGenes( api = cgds,  
  studyId = "gbm_tcga_pub",  
  genes = c("NF1", "TP53", "ABL1"),  
  by = "hugoGeneSymbol",  
  molecularProfileIds = "gbm_tcga_pub_mrna"  
)
## End(Not run)
```

---

**Node_obj_FreqIn**  
*Attribute interaction frequency to node size*

**Description**  
Attribute interaction frequency to node size

**Usage**  
Node_obj_FreqIn(geneList)
Node_obj_Met_ProfData

Arguments

geneList A list of gene symbol

Value

A data frame with node attributes

Examples

r_data <- new.env()
r_data[["FreqIn"]]<-structure(list(Genes = c("ATM", "ATR", "BRCA1", "BRCA2", "CHEK1", "CHEK2", "FANCF", "MDC1", "RAD51"), FreqSum = c(0.04, 0.05, 0.05, 0.03, 0.05, 0.04, 0.03, 0.03, 0.02)), .Names = c("Genes", "FreqSum"), class = "data.frame", row.names = c(NA, -9L))
## Not run:
GeneList <- whichGeneList("DNA_damage_Response")
nodeObj <- Node_obj_FreqIn(GeneList)
## End(Not run)

Node_obj_Met_ProfData  Attribute gene Methylation to Nodes

Description

Attribute gene Methylation to Nodes

Usage

Node_obj_Met_ProfData(list, type, threshold)

Arguments

list a list of data frame with methylation data

type HM450 or HM27

threshold the Rate cases (patients) that have a silencing genes by methylation

Value

a data frame with node shape attributes
Node_obj_mRNA_Classifier

**Examples**

cgds <- cBioPortal(
    hostname = "www.cbioportal.org",
    protocol = "https",
    api = "/api/v2/api-docs"
)
## Not run:
getDataByGenes( api = cgds,
    studyId = "gbm_tcga_pub",
    genes = c("NF1", "TP53", "ABL1"),
    by = "hugoGeneSymbol",
    molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)

---

**Node_obj_mRNA_Classifier**

Attribute genes expression to color nodes

**Description**

Attribute genes expression to color nodes

**Usage**

Node_obj_mRNA_Classifier(geneList, genesclassdetails)

**Arguments**

- **geneList**
  A gene list.
- **genesclassdetails**
  A dataframe with genes classes and genes expression.

**Value**

A data frame with node color attributes

**Examples**

r_data <- new.env()
input <- NULL

r_data[["FreqIn"]]<-structure(list(Genes = c("ATM", "ATR", "BRCA1", "BRCA2", "CHEK1",
    "CHEK2", "FANCF", "MDC1", "RAD51"), FreqSum = c(0.04, 0.05, 0.05,
    0.03, 0.05, 0.04, 0.03, 0.03, 0.02)), .Names = c("Genes", "FreqSum"),
    class = "data.frame", row.names = c(NA, -9L))
pickGO

Cleans up result from org.Xx.egGO and returns specific GO identifiers

Description
Cleans up result from org.Xx.egGO and returns GO identifier for either biological process (BP), cellular component (CC), or molecular function (MF). Can be used on list of GOs from translate, or a single list of GOs from an annotation package. May reduce list, if the (sub)list does not contain the chosen class!

Usage
pickGO(l, evidence = NA, category = NA)

Arguments

1  Character vector, or list of GO identifiers.

evidence  Character vector, filters on which kind of evidence to return; for a larger list see getEvidenceCodes. Evidence codes may be: c('IMP','IGI','IPI','ISS','IDA','IEP','IEA','TAS','NAS','ND','IC'). Leave as NA to ignore filtering on this part.

category  Character vector, filters on which ontology to return: biological process (BP), cellular component (CC), or molecular function (MF). Leave as NA to ignore filtering on this part.

Value
List with only the picked elements.

Author(s)
Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>
See Also

pickRefSeq, getEvidenceCodes, translate

Examples

library(org.Bt.eg.db)
genes <- c(280705, 280706, 100327208)
translate(genes, org.Bt.egSYMBOL)

symbols <- c("SERPINA1", "KERA", "CDS")
refseq <- translate(symbols, from=org.Bt.egSYMBOLEG, to=org.Bt.egREFSEQ)
# Pick the proteins:
pickRefSeq(refseq, priorities=c('NP', 'XP'), reduce='all')

# If you wanted do do some further mapping on the result from
# translate, simply use lapply.

library(GO.db)
GO <- translate(genes, org.Bt.egGO)
# Get all biological processes:
## Not run:
pickGO(GO, category='BP')
## End(Not run)

Description

When translating to RefSeq, typically multiple identifiers are returned, referring to different types of products, such as genomic molecule, mature mRNA or the protein, and they can be predicted, properties that can be read from the prefix (https://www.ncbi.nlm.nih.gov/refseq/). E.g. "XM_" is predicted mRNA and "NP_" is a protein. Run ?org.Bt.egREFSEQ.

Usage

pickRefSeq(
  1,
priorities = c("NP", "XP", "NM", "XM"),
reduce = c("all", "first", "last")
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Vector</td>
<td>Vector or list of RefSeqs accessions to pick from. If list given, applies the prioritisation to each element in the list.</td>
</tr>
<tr>
<td>priorities</td>
<td>Character vector of prioritised prefixes to pick by. Eg. c(&quot;NP&quot;, &quot;NM&quot;) returns RefSeqs starting 'NP', and if none found, those starting 'NM'. If no RefSeqs are found according to the priorities, Null is returned, unless the last element in priorities is '*'. Uses grepl, so see these for pattern matching. Default: c('NP','XP','NM','XM')</td>
</tr>
<tr>
<td>reduce</td>
<td>Reducing method, either return all annotations (one-to-many relation) or the first or last found annotation. The reducing step is applied after translating to the goal: all: returns all annotations first or last: choose first or last of arbitrarily ordered list.</td>
</tr>
</tbody>
</table>

Value

If vector given, returns vector. If list given, returns list without element where nothing could be picked.

Author(s)

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

Examples

```r
library(org.Bt.eg.db)
symbols <- c("SERPINA1", "KERA", "CDS")
refseq <- translate(symbols, from=org.Bt.egSYMBOL2EG, to=org.Bt.egREFSEQ)
mRNA <- pickRefSeq(refseq, priorities=c('NM','XM'))
proteins <- pickRefSeq(refseq, priorities=c('NP','XP'))
```

Description

Removes entries equal NA from list or vector.

Usage

```r
removeNAs(l)
```
renderCoffeewheel

Arguments

l Vector or list.

Author(s)

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

Examples

removeNAs(list('a'=NA, 'b'=c(NA, 'B'), 'c'='C'))

renderCoffeewheel Widget render function for use in Shiny

Description

Widget render function for use in Shiny

Usage

renderCoffeewheel(expr,env = parent.frame(), quoted = FALSE)

Arguments

expr id
env parent.frame()
quoted FALSE

Value


Examples

How <- "runManually"
## Not run:
coffeewheel(treeData = sampleWheelData)
## End(Not run)
renderMetabologram  Widget render function for use in Shiny

Description
Widget render function for use in Shiny

Usage
renderMetabologram(expr, env= parent.frame(), quoted = FALSE)

Arguments
  expr                expression
  env                 parent.frame()
  quoted             FALSE

Value
A circular plot with genetic profile in Shiny App.

Examples
## Not run:
library(bioCancer)
bioCancer::metabologram(treeData = sampleMetabologramData)

## End(Not run)

reStrColorGene  Restructure the list of color attributed to the genes in every dimension for every studies

Description
Restructure the list of color attributed to the genes in every dimension for every studies

Usage
reStrColorGene(df)

Arguments
df                data frame with colors attributed to the genes
Value
Hierarchical color attribute: gene > color

Examples
```r
cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)
## Not run:
dataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
  molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)
```

Description
Restructure the list of color attributed to the genes in every study for every dimensions

Usage
```r
reStrDimension(LIST)
```

Arguments
- **LIST** list of hierarchical dimensions

Value
Hierarchical structure of: Study > dimensions > gene > color

Examples
```r
cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)
## Not run:
dataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
  molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)
```
reStrDisease

Restructure the list of color attributed to the genes in every disease

Description
Restructure the list of color attributed to the genes in every disease

Usage
reStrDisease(List)

Arguments
List of data frame with color attributes

Value
Hierarchy of dimensions in the same study: dimensions > gene > color

Examples

```r
cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)
## Not run:
getDataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
  molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)
```
returnTextAreaInput  
*Return message when the filter formula is not correct (mRNA > 500)*

### Description
Return message when the filter formula is not correct (mRNA > 500)

### Usage
```r
returnTextAreaInput(inputId, 
  label = NULL, 
  rows = 2, 
  placeholder = NULL, 
  resize = "vertical", 
  value = "")
```

### Arguments
- **inputId**: The ID of the object
- **label**: Text describes the box area
- **rows**: Number of rows
- **placeholder**: Error message if needed
- **resize**: orientation of text
- **value**: default text in the area box

### Value
text message

### Examples
```r
ShinyApp <- 1
## Not run:
returnTextAreaInput(inputId = "data-filter", 
  label = "Error message", 
  rows = 2, 
  placeholder = "Provide a filter (e.g., Genes == 'ATM') and press return", 
  resize = "vertical", 
  value = "")
## End(Not run)
```
Studies_obj

get object for grViz. Link Studies to genes

Description

get object for grViz. Link Studies to genes

Usage

Studies_obj(df)

Arguments

df data frame with gene classes

Value

grViz object. a data frame with Study attributes

Examples

Studies_obj(data.frame("col1", "col2", "col3", "col4", "col5", "col6"))

## Not run:
Genes ranking class postProb exprsMeanDiff exprsUpDw
1 FANCF 1 brca_tcga 1.00000 179.9226 UP
2 MLH1 1 gbm_tcga 0.99703 256.3173 UP

## End(Not run)

switchButton

A function to change the Original checkbox of rshiny into a nice true/false or on/off switch button No javascript involved. Only CSS code.

Description

To be used with CSS script 'button.css' stored in a 'www' folder in your Shiny app folder

Usage

switchButton(inputId, label = NULL, value = FALSE, col = "GB", type = "TF")
Arguments

- **inputId**: The input slot that will be used to access the value.
- **label**: Display label for the control, or NULL for no label.
- **value**: Initial value (TRUE or FALSE).
- **col**: Color set of the switch button. Choose between "GB" (Grey-Blue) and "RG" (Red-Green)
- **type**: Text type of the button. Choose between "TF" (TRUE - FALSE), "OO" (ON - OFF) or leave empty for no text.

---

**test.CGDS**  
*S3 method to test cBioPortal connection*

---

**Description**

S3 method to test cBioPortal connection

**Usage**

```r
## S3 method for class 'CGDS'
test(x, ...)
```

**Arguments**

- **x**: connection object
- **...**: not used

---

**translate**  
*Translate between different identifiers*

---

**Description**

Function for translating from one annotation to another, eg. from RefSeq to Ensemble. This function takes a vector of annotation values and translates first to the primary annotation in the Biocore Data Team package (ie. entrez gene identifier for org.Bt.eg.db) and then to the desired product, while removing non-translated annotations and optionally reducing the result so there is only a one-to-one relation.
translate

translate(
  values,
  from,
  to = NULL,
  reduce = c("all", "first", "last"),
  return.list = TRUE,
  remove.missing = TRUE,
  simplify = FALSE,
  ...
)

Arguments

values  Vector of annotations that needs translation. Coerced to character vector.
from    Type of annotation values are given in. NB! take care in the orientation of the
         package, ie. if you have RefSeq annotations, use org.Bt.egREFSEQ2EG or (in some
cases) revmap(org.Bt.egREFSEQ).
to      Desired goal, eg. org.Bt.egENSEMBLPROT. If NULL (default), goal if the pack-
         ages primary annotation (eg. entrez gene for org.Bt,eg.db). Throws a warning if
         the organisms in from and to are not the same.
reduce  Reducing method, either return all annotations (one-to-many relation) or the
         first or last found annotation. The reducing step is applied after translating to
         the goal: all: returns all annotations first or last: choose first or last of arbitrarily
         ordered list.
return.list Logical, when TRUE, returns the translation as a list where names
remove.missing Logical, whether to remove non-translated values, defaults TRUE.
simplify Logical, unlists the result. Defaults to FALSE. Usefull when using translate
         in a lapply or sapply.
...     Additional arguments sent to pickGO if from returns GO set.

Details

If you want to do some further mapping on the result, you will have to use either unlist og lapply,
where the first returns all the end-products of the first mapping, returning a new list, and the latter
produces a list-within-list.

If from returns GO identifiers (e.g. from = org.Bt.egGO), then the returned resultset is more com-
plex and consists of several layers of lists instead of the usual list of character vectors. If to has
also been specified, the GO IDs must be extracted (internally) and you have the option of filtering
for evidence and category at this point. See pickGO.

Value

List; names of elements are values and the elements are the translated elements, or NULL if not
translatable with remove.missing = TRUE.
Note

Requires user to deliver the annotation packages such as org.Bt.egREFSEQ.

Author(s)

Stefan McKinnon Edwards <stefan.hoj-edwards@agrsci.dk>

See Also

pickRefSeq, pickGO

Examples

library(org.Bt.eg.db)
genes <- c(280705, 280706, 100327208)
translate(genes, org.Bt.egSYMBOL)

symbols <- c("SERPINA1","KERA","CD5")
refseq <- translate(symbols, from=org.Bt.egSYMBOL2EG, to=org.Bt.egREFSEQ)
# Pick the proteins:
pickRefSeq(refseq, priorities=c('NP','XP'), reduce='all')

# If you wanted do do some further mapping on the result from
# translate, simply use lapply.

library(GO.db)
GO <- translate(genes, org.Bt.egGO)

UnifyRowNames

Unify row names in data frame with the same order of gene list.

Description

Unify row names in data frame with the same order of gene list.

Usage

UnifyRowNames(x, geneList)

Arguments

x data frame with gene symbol in the row name
geneList a gene list

Value

a data frame having the gene in row name ordered as in gene list.
Examples

cgds <- cBioPortal(
  hostname = "www.cbioportal.org",
  protocol = "https",
  api = "/api/v2/api-docs"
)
## Not run:
getDataByGenes( api = cgds,
  studyId = "gbm_tcga_pub",
  genes = c("NF1", "TP53", "ABL1"),
  by = "hugoGeneSymbol",
  molecularProfileIds = "gbm_tcga_pub_mrna"
)
## End(Not run)

user_CNA

Example of Copy Number Alteration (CNA) dataset

Description

Example of Copy Number Alteration (CNA) dataset

Usage

user_CNA

Format

An object of class data.frame with 579 rows and 13 columns.

Author(s)

Karim Mezhoud <kmezhoud@gmail.com>

user_MetHM27

Example of Methylation HM27 dataset

Description

Example of Methylation HM27 dataset

Usage

user_MetHM27
user_MetHM450

Format
An object of class data.frame with 600 rows and 13 columns.

Author(s)
Karim Mezhoud <kmezhoud@gmail.com>

user_MetHM450  Example of Methylation HM450 dataset

Description
Example of Methylation HM450 dataset

Usage
user_MetHM450

Format
An object of class data.frame with 10 rows and 13 columns.

Author(s)
Karim Mezhoud <kmezhoud@gmail.com>

user_mRNA  Example of mRNA expression dataset

Description
Example of mRNA expression dataset

Usage
user_mRNA

Format
An object of class data.frame with 307 rows and 13 columns.

Author(s)
Karim Mezhoud <kmezhoud@gmail.com>
**user_Mut**

*Example of Mutation dataset*

**Description**

Example of Mutation dataset

**Usage**

user_Mut

**Format**

An object of class `data.frame` with 37 rows and 23 columns.

**Author(s)**

Karim Mezhoud <kmezhoud@gmail.com>

**whichGeneList**

*Verify which gene list is selected*

**Description**

Verify which gene list is selected

**Usage**

`whichGeneList(geneListLabel)`

**Arguments**

- `geneListLabel`  
The label of GeneList. There are three cases: "Genes" user gene list, "Reactome_GeneList" GeneList plus genes from reactomeFI "file name" from Examples

**Value**

Gene List label

**Examples**

```r
# How <- "runManually"
## Not run:
whichGeneList("102")

## End(Not run)
```
Description

Capture html output widget as .png in R

Usage

widgetThumbnail(p, thumbName, width = 1024, height = 1024)

Arguments

p is the html widget
thumbName is the name of the new png file
width 1024
height 1024

Value

3 files .html, .js and .png

Examples

How <- "runManually"
## Not run:
# Load package
library(networkD3)
library(htmlwidgets)
# Create fake data
networkData <- data.frame(src, target)
# Plot
plot = simpleNetwork(networkData)
# Save html as png
widgetThumbnail(p = plot, thumbName = "plot", width = 1024, height = 1024)
## End(Not run)
Index

* datasets
  epiGenomics, 15
  user_CNA, 44
  user_MetHM27, 44
  user_MetHM450, 45
  user_mRNA, 45
  user_Mut, 46

* package
  AnnotationFuncs, 5
  -package (bioCancer), 10
  .dbEscapeString, 3
  .getTableTableName, 4, 22
  .pickRef, 4

  AnnotationFuncs, 5
  attriColorGene, 6
  attriColorValue, 7
  attriColorVector, 8
  attriShape2Gene, 9
  attriShape2Node, 9

  bioCancer, 10

  CGDS, 10
  checkDimensions, 11
  coffeewheel, 12
  coffeewheelOutput, 12

  displayTable, 13

  Edges_Diseases_obj, 14
  epiGenomics, 15

  findPhantom, 15

  getEvidenceCodes, 16, 32, 33
  getFreqMutData, 16
  getGenesClassification, 17
  getList_Cases, 19
  getList_GenProfs, 20
  getListProfData, 18

  getOrthologs, 5, 6, 20
  getProfData, 22
  getSequenced_SampleSize, 23

  mapLists, 22, 24, 34
  metabologram, 25
  metabologramOutput, 26
  Mutation_obj, 26

  Node_df_FreqIn, 27
  Node_Diseases_obj, 28
  Node_obj_CNA_ProfData, 29
  Node_obj_FreqIn, 29
  Node_obj_Met_ProfData, 30
  Node_obj_mRNA_Classifier, 31

  pickGO, 16, 32, 42, 43
  pickRefSeq, 4, 5, 33, 33, 43

  removeNAs, 24, 34
  renderCoffeewheel, 35
  renderMetabologram, 36
  reStrColorGene, 36
  reStrDimension, 37
  reStrDisease, 38
  returnTextAreaInput, 39

  Studies_obj, 40
  switchButton, 40

  test.CGDS, 41
  translate, 5, 6, 21, 22, 32, 33, 41

  UnifyRowNames, 43
  user_CNA, 44
  user_MetHM27, 44
  user_MetHM450, 45
  user_mRNA, 45
  user_Mut, 46

  whichGeneList, 46
  widgetThumbnail, 47